

## A SENDER INITIATED RECOVERY ALGORITHM (SIRA) FOR THE LAYER 2 TUNNELING PROTOCOL (L2TP)

### CROSS-REFERENCE TO RELATED APPLICATIONS

Sub AP  
Related subject matter is disclosed in the co-pending, commonly assigned, U.S. Patent application of Chuah, entitled "A Receiver Initiated Recovery Algorithm (RIRA) for the Layer 2 Tunneling Protocol (L2TP)," Application No. XXX, filed on XXX.

### BACKGROUND OF THE INVENTION

#### (1) FIELD OF THE INVENTION

10 This invention relates generally to communications and, more particularly, to packet communications systems.

#### (2) BACKGROUND ART

568040 T 2564660  
The Layer Two Tunneling Protocol (L2TP) (e.g., see K. Hamzeh, T. Kolar, M. Littlewood, G. Singh Pall, J. Taarud, A. J. Valencia, W. Verthein, W.M. Townsley, B. Palter, A. Rubens "Layer Two Tunneling Protocol (L2TP)", Internet draft, March, 1998) is designed by the Internet Engineering Task Force (IETF) L2TP Working Group to allow internet service providers (ISP) to offer services other than traditional registered Internet Protocol (IP) address-based services. For example, ISPs can now offer virtual dial-up services to their customers via L2TP tunnels (or L2TP connections) allowing them to access corporate intranets.

20 There are two types of sessions, in an L2TP connection, namely a control session and a data session. For a control session, L2TP defines a retransmission scheme for control messages (also known as control packets) lost during transmission. However, L2TP does not retransmit lost payload messages (also known as payload packets) for a data session. Instead, when payload packets are missing, L2TP defines a sender-initiated recovery algorithm (SIRA) for resetting the "next received" (*Nr*) sequence number at the  
25 receiver. In particular, when the transmit window times-out (i.e., the sender has transmitted a predefined number of packets without receiving an acknowledgement of a

earlier-sent packet) the sender transmits a payload message that includes a predefined "Reset *Sr*" (*R-bit*) indicator, which resets the value for *Nr* (at the receiver) to either just beyond the first missing packet or to the current send sequence number of the sender.

## SUMMARY OF THE INVENTION

5 In accordance with the invention, a Layer 2 Tunneling Protocol (L2TP) sender sends packets to an L2TP receiver and initiates a recovery process upon detection of a multiple messages from the L2TP receiver that the L2TP receiver is still waiting for a prior transmitted packet.

10 In an embodiment of the invention, an L2TP connection is established between two packet interfaces, i.e., an L2TP Access Concentrator (LAC) and a L2TP Network Server (LNS). The sender of at least one of these each packet interfaces performs a sender initiated recovery algorithm (SIRA) upon detection of multiple messages from the L2TP receiver that the L2TP receiver is still waiting for a prior transmitted packet. In particular, if the L2TP sender receives a predefined number of packets including the same  
15 "next received" (*Nr*) sequence number, the L2TP sender transmits a payload message that includes the "Reset *Sr*" (*R-bit*) indicator, which resets the value for *Nr* (at the receiver) to either just beyond the first missing packet or to the current send sequence number of the sender.

## BRIEF DESCRIPTION OF THE DRAWING

20 FIG. 1 shows an L2TP prior art transmission sequence;

FIG. 2 shows a communications system in accordance with the principles of the invention;

FIGs. 3, 4 and 5 show illustrative flow charts of methods in accordance with the principles of the invention; and

25 FIG. 6 shows a block diagram of an illustrative packet interface embodying the principles of the invention.

## DETAILED DESCRIPTION

At this point, before describing the inventive concept, a brief review is provided of

L2TP sequence numbers. If the reader is familiar with this background, simply skip-ahead to the section entitled "Sender Initiated Recovery Algorithm (SIRA)."

### L2TP Sequence Numbers

In L2TP, each peer maintains a number of sequence number states. A single  
 5 sequence number state is maintained for all control messages, and a session-specific  
 sequence number state is maintained for the payload of each user session within a tunnel.  
 For example, assume an L2TP Access Concentrator (LAC) has established an L2TP  
 tunnel with an L2TP Network Server (LNS) such that the L2TP tunnel conveys two user  
 sessions. The LAC then maintains one sequence number state for control messages to the  
 10 LNS and two session-specific sequence number states, one for each user session.  
 Similarly, the LNS also maintains its own sequence number state for control messages and  
 two session-specific sequence number states. Further, the sender and receiver negotiate a  
 transmit window size (in packets) that represents the number of packets the sender may  
 transmit before requiring an acknowledgement from the receiver for an earlier transmitted  
 15 packet (described further below).

A sequence number state is represented by distinct pair of state variables,  $Sr$  and  
 $Ss$ .  $Sr$  represents the expected sequence value of the next message from a peer.  $Ss$   
 represents the sequence value of the  $Ns$  field (described below) for the next message to be  
 sent to the opposite peer. Each state,  $Sr$  and  $Ss$ , is initialized such that the first message  
 20 sent and the first message expected to be received for each session has an  $Ns$  value of 0.  
 This corresponds to initializing  $Ss$  and  $Sr$  in both peers to 0 for each new session.

In L2TP, there are two fields for use in a packet: the  $Nr$  (Next Received) field and  
 $Ns$  (Next Sent) field. These two fields are always present in control messages, and are  
 optionally present in payload packets. Every time a peer sends a non-zero length message,  
 25 it increments its corresponding  $Ss$  value for that session by 1. This increment takes place  
 after the current  $Ss$  value is copied to  $Ns$  in the message to be sent. Outgoing messages  
 always include the current value of  $Sr$  for the corresponding session in their  $Nr$  field  
 (however, if sent before any packet is received,  $Nr$  is 0). When a non zero-length message  
 is received with an  $Ns$  value that matches the session's current  $Sr$  value,  $Sr$  is incremented  
 30 by 1. It should be noted that, for both control and payload sessions,  $Sr$  is not modified if a

message is received with a value of  $Ns$  greater than the current  $Sr$  value.

Upon receipt of an in-order non-zero-length message, the receiving peer must acknowledge the message by sending back the updated value of  $Sr$  in the  $Nr$  field of the next outgoing message. This updated  $Sr$  value can be piggybacked in the  $Nr$  field of any non-zero-length outgoing messages that the peer may happen to send back.

A message (control or payload) with a zero-length body indicates that the packet is only used to communicate  $Nr$  and  $Ns$  fields. The  $Nr$  and  $Ns$  fields are filled in as described above, but the sequence number state,  $Ss$ , is not incremented. Thus, a zero-length message sent after a non-zero-length message contain a new  $Ns$  value while a non-zero-length message sent after a zero-length message contains the same value of  $Ns$  as the preceding zero-length message.

If the peer does not have a message to transmit after receiving a non-zero-length message, then it should send a zero-length message containing the latest values of  $Sr$  and  $Ss$ , as described above upon expiration of a timeout. The suggested value for this timeout is  $1/4$  of the round trip time (RTT), if computed by the receiving peer, or a maximum of  $1/2$  second otherwise. This timeout should provide a reasonable opportunity for the receiving peer to obtain a payload message destined for its peer, upon which the ACK of the received message can be piggybacked. (This timeout value should be treated as a suggested maximum. To provide better throughput, the receiving peer should skip this timeout entirely and send a zero-length message immediately in the case where its receive window fills and it has no queued data to send for this connection or it can't send queued data because the transmit window is closed.)

Upon expiration of the timeout,  $Sr$  is compared to  $Lr$  (the last sent  $Nr$ ) and if they are not equal, a zero-length ACK is issued. If they are equal, then no ACKs are outstanding and no action needs to be taken. The timer should not be reinitialized if a new message is received while it is active since such messages will be acknowledged when the timeout expires. This ensures that periodic ACKs are issued with a maximum period equal to the recommended timeout interval. This interval should be short enough to not cause false acknowledgement timeouts at the transmitter when payload messages are being sent in one direction only. Since such ACKs are being sent on what would otherwise be an idle

data path, their affect on performance should be small, if not negligible.

For a control session, if outgoing messages are lost, retransmission of outgoing messages should eventually provide the receiving peer with the expected message. For payload sessions, however, lost messages are never retransmitted.

5        When a sent message is lost, the  $Sr$  value at the receiving peer sticks at the  $Ns$  value of the first missing payload message. In L2TP, a Sender Initiated Recovery Algorithm (SIRA) is defined and illustrated in FIG. 1. In SIRA, the sender uses a timer mechanism to determine when transmitted payload packets have been lost. This timer mechanism uses the above-mentioned transmit window (and is also referred to as a form of "window time-out") and a predefined time-out value  $T$ . With respect to FIG. 1, it is assumed that the window size is 4 packets. Initially, the L2TP sender transmits a packet #1 (i.e., representative of a sequence number 1) to an L2TP receiver. Upon receipt, the L2TP receiver responds by sending a packet that includes an  $Nr$  value of #2 as the value of the next in-sequence packet the receiver expects to receive. The L2TP sender then transmits a packet #2. However, as shown in FIG. 1, this packet is not received by the L2TP receiver. Subsequently, the L2TP sender transmits packet #3, to which the L2TP receiver responds with a packet that includes an  $Nr$  value of #2. As such, the L2TP sender has not yet received a positive acknowledgement that packet #2 was received by the LT2P receiver (i.e., this is a negative acknowledgement). As is illustrated in FIG. 1, eventually, the L2TP sender transmits 4 packets (#2, #3, #4, and #5) without receiving a positive acknowledgement of packet #2. As such, the window size of 4 packets has been reached without receiving a positive acknowledgement (i.e., a window time-out occurs) and the L2TP sender halts transmission for a period of time  $T$ . If no positive acknowledgement is received before the period of time  $T$  expires, the L2TP sender transmits a payload message that includes a predefined "Reset  $Sr$ " ( $R-bit$ ) indicator. Upon detection by the receiving peer of the  $R-bit$  indicator, the receiving peer updates its  $Sr$  value to the value of  $Ns$  contained in the received payload message (that included the  $R-bit$  indicator) if  $Ns > Sr$ . In other words, the  $R-bit$  indicator is used to reset the  $Sr$  value to the value of  $Ns$ . The sender can determine whether it wants to reset the  $Sr$  value of the receiver to one past the first missing payload message or to the current value of  $Ss$  as the

sender. Unless the *R-bit* is set, a peer receiving a zero-length message does not update its *Sr* variable.

*Sender Initiated Recovery Algorithm (SIRA)*

In accordance with the invention, a Layer 2 Tunneling Protocol (L2TP) sender  
5 sends packets to an L2TP receiver and initiates a recovery process upon detection of a multiple messages from the L2TP receiver that the L2TP receiver is still waiting for a prior transmitted packet.

FIG. 2 shows an illustrative communications system 100 in accordance with the principles of the invention. Other than the inventive concept, the elements are well-known  
10 and will not be described in detail. For example, personal computer (PC) 105 includes data communications equipment (not shown) for dial-up access through public-switched-network (PSTN) 110 to ISP A for establishing an Internet connection. Likewise, the solid lines between elements of communications system 100 are representative of well-known communications facilities between the respective endpoints, e.g., the connection between  
15 PC 105 and PSTN 110 is representative of a local loop connection, the connection between ISP A and Internet 130 is supported by asynchronous transfer mode (ATM) over a synchronous optical network (SONET), etc.

As can be observed from FIG. 2, communications system 100 further comprises ISP A, as represented by ISP A Network. The latter comprises network access server  
20 (NAS) 155 (also referred to herein as an L2TP Access Concentrator (LAC)), which includes a point-of-presence (POP) router (not shown) as known in the art, a local network 160, and a router 165. It is assumed that ISP A provides a virtual private network (VPN) service for remotely located employees (associated with PC 105) to access an illustrative Corporate Network via network server (NS) 135 (also referred to  
25 herein as an L2TP Network Server (LNS)), which provides, among other functions, a routing and firewall capability. (The Corporate network is assumed to be, e.g., a collection of local area networks (not shown) suitably protected behind the LNS.) For the purposes of this description, it is assumed that the reader is familiar with the above-mentioned L2TP protocol.

30 Reference should also be made at this time to FIG. 3, which shows an illustrative

flow chart of a method in accordance with the principles of the invention. For the purposes of this description, the inventive concept is described in the context of an illustrative packet interface, e.g., the LAC, in the context of sending packets to the LNS. However, it should be understood that the inventive concept is not so limited and equally applies to any packet interface that sends packets, such as the LNS. (It is presumed that the LAC and the other respective servers are suitably programmed to carry out the below-described methods using conventional programming techniques, which, as such, will not be described herein.)

In step 205, the LAC establishes an L2TP connection 2 with the LNS for communicating data between PC 110 and the LNS, which provides a gateway to the corporate network (not shown in detail). As known in the art, step 205, either results in the creation of a new tunnel to support the call between the user (not shown) associated with PC 110 and the LNA, or simply adds an L2TP connection to an existing L2TP tunnel. (Although not described herein, it is assumed that there is a point-to-point (PPP) connection 1 between the LAC and PC 105. That is, payload packets over L2TP connection 2 are L2TP encapsulated PPP packets for user sessions between the LAC/LNS pair.)

As noted above, there are two types of L2TP sessions in an L2TP connection, a control session and a payload session. (It should also be noted that the term control connection and payload connection may also be used, respectively.) In accordance with the invention, in step 210, after establishing an L2TP connection, an L2TP sender (here, illustrated by the LAC) initiates a sender initiated recovery algorithm (SIRA) upon detection that a predefined number of packets have been received that include the same "next received" ( $Nr$ ) sequence number (described below). The LAC continues to perform the SIRA if necessary until the L2TP connection is disconnected in step 215.

Turning now to FIG. 4, another flow chart is shown further illustrating the SIRA of step 210. At the start of this process, it is assumed that the LAC stores a variable "multiple-negative-acknowledgements" and initializes this variable to zero. It is assumed this variable is a simple data structure that associates a sequence number value  $ONr$  (outstanding sequence number) with a counter value,  $K$ . (It should be noted that, as

known in the art, sequence numbers are typically repeated, e.g., modulo 8, etc. (for modulo 8, the sequence numbers, 0, 1, ..., 6, 7, are used). As such, the below described method presumes that the LAC takes into account the eventual re-use of a sequence number.) In step 305, the LAC receives a packet that includes an  $Nr$  value representing the next sequence number packet expected by the LNS (receiver). In step 310, the LAC checks for an outstanding packet, e.g., is the received  $Nr$  value the same as the current value of  $ONr$ . If the values are different, i.e., there is no outstanding packet, the LAC updates the variables in step 325, e.g., the LAC updates the value of  $ONr$  to the received  $Nr$  value, sets the value of  $K$  to 1, and continues to step 305 to receive packets. However, if there is an outstanding packet, e.g., the received value for  $Nr$  is equal to the value for  $ONr$  (the same packet),  $K$  is incremented by 1 in step 315 and the LAC checks if a *predefined limit* has been reached in step 320. (The value of the *predefined limit* is determined empirically. For illustration, it is assumed that the *predefined limit* is 3.) If, in step 320, less than 3 multiple negative acknowledgments have been received, then the LAC continues to step 305 to receive packets. However, if the limit has been reached in step 315, then the LAC transmits a payload message (in step 330) to the LNS that includes the "Reset  $Sr$ " (*R-bit*) indicator, which resets the value for  $Nr$  (at the receiver) to either just beyond the first missing packet or to the current send sequence number of the sender. In addition, the LAC re-initializes the "*multiple-negative-acknowledgements*" variable and continues to step 305 to receive packets.

It should be noted that, setting a window size to be two or three times the round trip times is sufficient to give relatively high throughput. In addition, the timeout interval can be chosen to be comparable to round trip time plus two or four standard deviations. Finally, it is still required that the sender transmits an *R-bit* packet when the window of the sender is closed for more than a window timeout value. Illustratively, a buffer (not shown) of size of 128 packets is assumed. However, buffer size only needs to be one more than the window size. An illustrative window size is 7 packets. An illustrative RTT is 4 packets.

In accordance with a feature of the invention, the sender can also incorporate a "*payload time-out*" feature. In the *payload time-out* feature, if a predefined period of



time,  $\tau$ , expires before receiving a positive acknowledgement from the receiver, the sender transmits a payload message to the receiver that includes the "Reset  $Sr$ " (*R-bit*) indicator, which resets the value for  $Nr$  (at the receiver) to either just beyond the first missing packet or to the current send sequence number of the sender. A flow chart for the *payload time-out* feature is shown in FIG. 5. An illustrative value for a payload timeout is the RTT.

Turning briefly to FIG. 6, a high-level block diagram of a representative packet interface for use in performing SIRA in accordance with the principles of the invention is shown. Packet interface 605 is a stored-program-control based processor architecture and includes processor 650, memory 660 (for storing program instructions and data, e.g., for performing the illustrative methods shown in FIGs. 3, 4, and 5, etc.) and communications interface(s) 665 for coupling to one or more packet communication facilities as represented by path 666.

The foregoing merely illustrates the principles of the invention and it will thus be appreciated that those skilled in the art will be able to devise numerous alternative arrangements which, although not explicitly described herein, embody the principles of the invention and are within its spirit and scope.